1. **js中基本类型有哪些？**

Boolean、null、string、undefined、number、symbol

1. **typeof是否能正确判断类型？**

Typeof对于原始类型来说，除了null都可以显示正确的类型

1. **怎么判断this？**

根据方法的调用者确认

1. **实现一个函数，第一次调用输出1，第二次2，以此类推...？**

闭包实现

1. **ES5和ES6继承实现？**

    // 使用ES5 创建类

    function Parent(name) {

      this.name = name

    }

    // 原型上定义方法

    Parent.prototype.getName = function () {

      console.log(this.name);

    }

    // 继承

    function Child(name) {

      // 使用call继承父类实例上的属性

      Parent.call(this, name)

    }

    // 定义静态属性

    Child.sta = 100

    // 使用Object.creacte继承原型属性和方法

    Child.prototype = Object.create(Parent.prototype, {

      // 设置子类构造器

      constructor: {

        value: Child

      }

    })

    var chil = new Child('小红')

    console.log(chil);

   // Es6 class关键字创建类

    class Par {

      constructor(name) {

        this.name = name

      }

      getName() {

        console.log(this.name);

      }

    }

    // Es6 继承：extends关键字

    class Chi extends Par {

      constructor(name) {

        super(name);

        this.age = 10;

        this.setAge = function () {

          this.age = 100

        }

      }

      // 定义属性

      other = 10

      //定义方法

      hander() {

        console.log(this.other);

      }

      // 定义静态属性：static关键字

      static sta = 10000

    }

    let Chi2 = new Chi('小明')

    console.log(Chi2);

1. **var、let及const区别？**

Var变量提升到作用域顶部

使用let和const，变量不会被挂载到window上，声明后才可以使用

1. **讲讲js模块化？**

好处：

解决命名冲突

提高复用性

提高代码可维护性

有哪些？

AMD和CMD

CommonJS

CommonJS最早是Node在使用，目前也仍然广泛使用，比如在webpack中

ESModule

ESModule是原生支持的模块化

1. **map，filter，reduce作用是啥？**

Map作用是生成一个新数组，遍历原数组，将每个元素拿出来做一些变换然后放入到新的数组中。

Filter的作用也是生成一个行数组，在遍历数组的时候将返回值为true的元素放入新数组。

Reduce可以将数组中的元素通过回调函数最终转为一个值。

1. **async和await的特点，它们的优点和缺点分别是什么？**

Async和await是异步编程的最优解。

①解决了promise我的回调地狱问题

②因为await将异步代码改造成了同步代码，如果多个异步代码没有依赖却使用了await会导致性能上的降低

1. **dom事件代理是什么？**

事件委托

1. **什么是跨域？**

服务器和后端

通信：原则=》协议=》标准

浏览器出于安全考虑，有同源策略。也就是说，如果协议、域名或者端口有一个不同就是跨域，ajax会失败。

目的：防止CSRF攻击

实际开发：

①CORS=》后端同学（api）设置请求allow=\*

②JSONP（get）=》scrpit标签支持跨域=》src=》访问地址+回调函数

③本地：Nginx

1. **MVVM是什么？**

View：用户看到的视图

Model：本地数据和数据库的数据

在mvvm架构中，引入了ViewModel的概念。ViewModel只关心数据和业务的处理，不关心View如何处理数据，在这种情况下，View和Model都可以独立出来，任何一方改变了也不一定需要改变另一方，并且可以将一些可复用的逻辑放在一个ViewModel中，让多个View复用这个ViewModel。

以Vue框架举例，ViewModel就是组件的实例。View就是模板，Model的就是data（）

1. **谈谈你对原型，原型链的理解**

每个构造函数一旦创建都有prototype指针，指向它的原型对象，而原型对象，会默认生成一个constructor指针又指向构造函数。

1. **什么是闭包，在哪里用过闭包，闭包的使用场景是什么**

闭包是什么？ 闭包就是能够读取其函数内部变量的函数

在哪用过闭包？ 需要获取函数内部变量的时候

闭包的使用场景是什么？ 闭包可以用在很多地方，它的最大用处有两个，一个是可以读取函数内部的变量，另一个就是让这些变量的值始终保持在内存中。

1. **怎么实现继承**

①原型链继承：将父类的实例作为子类的原型，继承父类方法

②构造继承：利用call或者apply把父类通过this指定的属性和方法复制到子类创建的实例中。

③组合继承：通过调用父类构造，继承父类的属性并保留传参的优点，然后再通过将父类实例作为子类原型，实际函数复用

1. **ajax的工作原理**

如果请求是get

①创建xhr对象 var xhr = new XMLHttpRequest（）

②设置请求的方式和路径 xhr.open（‘get’，请求的路径）

③发送请求 xhr.send（）

④注册事件，获取响应内容 xhr.onreadystatecheng = function（）

如果请求是post

①创建xhr对象var xhr = new XMLHttpRequest（）

②设置请求的方式和路径 xhr.open（‘post’，请求的路径）

③设置请求头xhr.setRequestHeader（告诉浏览器 发送过去的请求的格式）

④发送请求体 xhr.send（请求体）

⑤注册事件，接收响应内容 xhr.onreadystatechange = function（）

1. **vue-router怎么配置路由**

①router-link制作导航

②设置容器 router-view

③js提供要渲染组件选择对象

④js实例化路由 new VueRouter（{}）

⑤js配置路由 routers：[{}]

⑥路由挂载到new vue里

1. **怎么获取动态路由参数**

$route.params

1. **怎么解决spa首屏加载慢的问题**

懒加载：这种优化，就是将每个组件的js代码独立起来，在使用到这个组件时，才向服务器请求文件，并且请求过一次后就会缓存下来，再次使用这个组件时，就会使用缓存，不用发送请求。

CDN:CDN引入依赖文件

1. **vue全家桶是什么**

Vue-cli项目构建工具

Vue-router 路由

Vuex 状态管理

Axios http请求工具

Webpack